Postel

7142

Network Working Group
Request for Comments # 197
NIC # 7142
Categories: C.3, D.1
Updates: None
Obsoletes: None

A. Shoshani, SDC
E. Harslem, Rand
14 July 1971

INITIAL CONNECTION PROTOCOL--REVIEWED

INTRODUCTION

At the Network meeting preceding the SJCC '71, an "ICP Committee" was established. It's purpose was to get "something" working fast with minimum modifications to the current ICP so as to minimize complaints. (This seems like a good definition for almost everything!) Consequently, those who had objections to the current ICP were interviewed and a compromise was reached in the form of RFC #165. The ICP committee didn't have a chance to think about an alternative because of the above mentioned constraints. In this note we attempted a simple version of an ICP assuming that we can add commands to Host-Host protocol. We hope that this will be useful in the design of the next version of the Host-Host protocol.

ICP COMMANDS

To establish a regular connection one party can issue an INIT (NCP sends RTS or STR commands), then the other party can accept the request for connection by responding with an INIT or refusing it with a CLOSE. We think that

a similar, simple mechanism is desirable for the ICP. Furthermore, the ICP should allow for simplex as well as duplex connections from user to server.

The following commands are necessary for simplex connections:

ISC - Initiate Simplex Connection

ASC - Accept Simplex Connection

RSC - Refuse Simplex Connection

The notation for parameters is similar to that of RFC #165:

L - Server socket name, in one special case the server is "logger".

U - User socket.

S - Socket assigned by server for the connection with user.

X - Is the byte size if U is odd and is the link number if U is even.

$\overline{X}$ - Is the complement of X ($\overline{X}$ is the link number if U is odd and byte size if U is even.

To initiate a simplex connection the user's NCP issues:

ISC,L,U,X

To refuse this connection the server's NCP issues:

RSC,L,U

To accept this connection the server's NCP issues:

ASC,L,U,S,$\overline{X}$

Similarly, for duplex connections, we have:

IDC,L,U1,X1,U2,X2

RDC,L,U1,U2

ADC,L,U1,S1,$\overline{X1}$,U2,S2,$\overline{X2}$

where (U1,U2), (S1,S2), (U1,S1) and (U2,S2) are pairs of opposite gender.

After the server accepts the connection(s), it (they) goes immediately to a "connected state", and the appropriate ALL command(s) must be sent.

ADVANTAGES

The main advantage to this approach is that it minimizes the dialog between user and server. The server socket L is used only as an address, not a socket to connect to, therefore eliminating the need to establish a connection to L, choose a byte size, send an ALL command, send and receive data on it and CLOSE it. Race conditions as mentioned in RFC #143 do not arise. Socket L is the server and should be in a "Listening for ICP" state when an ISC or IDC is received. If socket L is not in that state, the serving NCP should refuse to ICP request. The serving NCP should not queue ICP requests.

In the current ICP, when the user choses socket U, he has to reserve sockets U+2 and U+3. In the above described approach no restrictions exist for U1 and U2 (except that they are of opposite gender); the same is true for S1 and S2.

We think that duplex commands are necessary since both connections are to be connected to the same server process. Their separation by using two ISCs, will add complications of correlating the two ISCs with the same process. Also, if two ISCs are used, the first might be accepted and the second refused. This leads to uncertainty on the user's part. This condition cannot occur with the duplex commands.

MINIMUM MODIFICATION TO CURRENT ICP

The minimum change we can think of to make the current ICP look similar to the above is to add one NCP level command -- accept:

ACC,L,U,S

The exchange between NCPs in the notation of RFC #165 is now:

Server NCP	User NCP
Listen for connection on L	RTS,U,L,ℓ_A
ACC,L,U,S	S is passed by NCP to the user and connection from U to L is closed.
STR,S+1,U+2,B_s	STR,U+3,S,B_u
RTS,S,U+3,ℓ_B	RTS,U+2,S+1,ℓ_c
Wait for connection	Wait for connection
ALL,ℓ_B,m_B,b_B	ALL,ℓ_c,m_c,b_c

An alternative way to the ACC command is a CLS command with an additional parameter (32 bits long). If parameter is zero the request for connection by the user is refused; if the parameter is non-zero, the request is accepted and socket S is the value of the parameter.

All suggested changes improve the ICP dialog both from the aesthetic and efficiency points of view. We lean strongly, however, to the first, more major ICP modification.

A COMMENT ABOUT CLS COMMAND

It seems appropriate to mention here for the purpose of the next version of the Host-Host protocol that the CLS command has more than one function. We think that the

CLS command should be reserved to close connections in the "connected state" only (i.e., "open" connections). Two additional commands can be used for "refusing" and "rejecting" requests for connections:

> REJ<mysocket><yoursocket> -- when a request for connection is rejected unconditionally.
>
> REF<mysocket><yoursocket><reason> -- when a request for connection is refused temporarily because the NCP could not handle it. For example: no process LISTENed to it and it was timed-out, or NCP tables are full in which case the user process may try again. The reason for refusing is indicated in the parameter "reason".